OpenMXとDFT②

尾崎 泰助
東京大学 物性研究所

OpenMXとDFT: Agenda①

- 本講義の対象者
- OpenMXの概要
- 開発・公開の経緯
- 開発・公開の意図
- 応用事例
- DFTソフトウエアの開発状況
- ソフトウエア公開のレベル
- de facto standardソフトウエアの条件
- プログラム開発において留意していること
- プログラムの全体構造
- デバッグの方法

OpenMXとDFT: Agenda②

- 安定性向上へのいくつかの工夫
- プログラム実装の品質テスト
- プログラム公開の際の心構え
- 公開は英語で
- マニュアル作成時に留意していること
- リリースの実際
- ユーザービリティ向上への工夫
- ユーザーサポートの実際
- ワークショップ・講習会の開催
- 共同開発におけるポリシー
- 共同開発の実際
- プログラムの保守管理
- 将来の発展へ向けたいくつの展望
- ソフトウエア開発は長期戦
- ソフトウエアを育てるために

安定性向上へのいくつかの工夫#1

長時間の計算後にプログラムが異常終了することはできるだけ避けたい。
この事態が頻繁に起こるソフトウエアに対し、ユーザーは不満を募らせ、
他のソフトウエアの使用を検討する。

- 入力パラメーターをチェックし、不適切な入力値の場合にはエラーメッセージを表示し、終了する。
- 入力パラメーターが与えられていない場合にはデフォルト値が自動設定される。
- 最適化レベル高のコンパイルオプションを設定した場合、計算が不安定になるルーチンがある。プログラム中で直接、コンパイルオプションのレベルを指定し、ユーザーの指定を受け付けない。
- 変数値が“nan”, “NaN”, “inf” , “Inf” となっていないか監視し、その場合には分岐して別処理を行い、計算が正常に進行するようにする。
- 複数の固有値ルーチンを用意しておき、on-the-flyで計算を監視して数値不安定性が検知された場合には他の固有値ルーチンに自動で乗り換える。

安定性向上へのいくつかの工夫#2

入力値のチェックとデフォルト値の設定

入力値を容易にチェックできるように入力補助関数群を作成

Inputtool.c & Inputtool.h

入力ファイル中でkeywordを探して変数に代入、なければデフォルト値を設定

```
input_string("System.CurrrentDirectory",filepath,"./");
input_int("level.of.stdout", &level_stdout,1);
input_logical("memory.usage.fileout",&memoryusage_fileout,0);
input_double("scf.energycutoff",&Grid_Ecut,(double)150.0);
 if (fp=input_find("<Atoms.Unitvectors")) {

    for (i=1; i<=3; i++){
      fscanf(fp,"%lf %lf %lf",&tv[i][1],&tv[i][2],&tv[i][3]);
    }
    if ( ! input_last("Atoms.Unitvectors>") ) {
      /* format error */
      printf("Format error for Atoms.Unitvectors¥n");
      po++;
    }
}
```

安定性向上へのいくつかの工夫#3

最適化オプションの直接設定

最適化レベル高のコンパイルオプションを設定した場合、計算が不安定になるルーチンがある。プログラム中で直接、コンパイルオプションのレベルを指定する。

```
#pragma optimization_level 1
void Allocation_Atoms_3D(int MD_iter, int weight_flag)
{.....
}
```

```
#pragma optimization_level 1
void XC_PBE(double dens[2], double GDENS[3][2], double Exc[2],
        double DEXDD[2], double DECDD[2],
        double DEXDGD[3][2], double DECDGD[3][2])
{.....
}
```

このプレ最適化オプション指定により、計算の安定性は大幅に向上する。

安定性向上へのいくつかの工夫#4

“nan”, “NaN”, “inf” , “Inf” の監視

変数値が“nan”, “NaN”, “inf” , “Inf” となっていないか監視し、その場合には分岐して別処理を行い、計算が正常に進行するようにする。

例えば、RMM-DIIS法による電子密度混合法は密度が収束してくると残差密度が線形従属に近づき、線形方程式がill-conditioningとなって計算が破綻する場合がある。

線形係数の値を監視し、値が”nan”、”NaN”、”inf”、”Inf”となっている場合には別処理を行い、計算が正常に進行するようにする。

監視を行わずに計算を続ければ、当然のことながら、計算は破綻し、異常終了する。

```
/************************************
  check "nan", "NaN", "inf" or "Inf"
************************************/

flag_nan = 0;
for (SCFi=1; SCFi<=NumMix; SCFi++){

  sprintf(nanchar,"%8.4f",alden[SCFi]);
  if (strstr(nanchar,"nan")!=NULL || strstr(nanchar,"NaN")!=NULL
   || strstr(nanchar,"inf")!=NULL || strstr(nanchar,"Inf")!=NULL){

    flag_nan = 1;
  }
}

if (flag_nan==1){
  for (SCFi=1; SCFi<=NumMix; SCFi++){
    alden[SCFi] = 0.0;
  }
  alden[1] = 0.1;
  alden[2] = 0.9;
}
```

安定性向上へのいくつかの工夫#5

固有値ソルバーのon-the-fly 選択

複数の固有値ルーチンを用意しておき、on-the-flyで計算を監視して数値不安性が検知された場合には他の固有値ルーチンに自動で乗り換える。

(1) これまでの数値実験の蓄積に基づき、行列サイズ毎に安定なルーチンを選択

```
if (n<10)
  EigenBand_lapack(ac, ko, n, n, 1);

else if (scf_eigen_lib_flag==0 || n<100)
  Eigen_Improved_PHH(MPI_Current_Comm_WD, ac, ko, n, EVmax, bcast_flag);

else if (scf_eigen_lib_flag==1)
  Eigen_ELPA1_Co(MPI_Current_Comm_WD, ac, ko, n, EVmax, bcast_flag);
```

(2) 対称性の高い系の場合、固有ベクトルの計算において数値不安定性が発生しやすい。固有ベクトルから状態数をカウントすることで、縮退系の固有ベクトル計算の精度が判断できる。精度が十分でない場合には三重対角行列のソルバーを変更する。

参考情報: vx → qr → dc → gr

```
Dnum = TZ - Total_Mul - system_charge;

  if (1.0e-8<fabs(Dnum)){

    if      (dste_flag==2) tmp_flag = 3;   /* vx -> qr    */
    else if (dste_flag==3) tmp_flag = 1;   /* qr -> dc    */
    else if (dste_flag==1) tmp_flag = 0;   /* dc -> gr    */
    else if (dste_flag==0){                /* gr -> ELPA1 */
      tmp_flag = 2;
      rediagonalize_flag_overlap_matrix_ELPA1 = 1;
    }

    dste_flag = tmp_flag;
```

プログラム実装の品質テスト#1

多岐に渡る機能を実装していくと、新しい機能の実装の際に、古い機能を壊してしまう場合がしばしばある。機能の追加に伴い、チェック項目が増加し、開発速度が著しく低下する。いかに効率よく実装の品質テストを行うべきか？

OpenMX本体の中に自動テスト機能を組み込み、容易に過去の実装した機能のチェックができるようにしてある。

(a) メモリリークテスタ

13種類の入力ファイルに対して実際の使用メモリの時系列変化をチェック。

(b) 力の計算テスタ

37種類の入力ファイルに対して力の数値解と解析解を比較。

(c) 小規模、中規模、大規模計算テスタ

37種類の入力ファイルに対して計算を実行し、参照値と比較。

(d) その他のテスタ

構造最適化テスタ、伝導計算テスタ、ワニア関数計算テスタ　等。

プログラム実装の品質テスト#2

メモリリークテスタ

構造を固定したまま構造最適化のループを反復し(SCF計算もその都度、行う)、プログラム中の定点でシステムコールしてVSZ(プロセスの仮想メモリサイズ)とRSS(実メモリ上の使用サイズ)を監視する。

オプションをつけてopenmxを実行するだけで良い。

% ./openmx -mltest

13種類の入力に対するメモリ使用量の観測結果はmltest.resultに出力される。

[ozaki@mx63 tmp]$ more mltest.result

1 ml_example/Co4.dat

	CPU (%)	VSZ (kbyte)	RSS (kbyte)
MD_iter= 1	97.700	254732	50588
MD_iter= 2	94.700	341748	171264
MD_iter= 3	98.800	335304	164944
MD_iter= 4	96.500	335304	164944
MD_iter= 5	94.900	335304	164944
MD_iter= 6	97.600	335304	164944
MD_iter= 7	96.100	335304	164944
MD_iter= 8	98.300	335304	164944
MD_iter= 9	97.000	335304	164944
MD_iter= 10	98.800	335304	164944
MD_iter= 11	97.600	335304	164944

プログラム実装の品質テスト#3

力の計算テスタ

シビアでかつ必ずクリアすべき品質テストは全エネルギーと力の計算が無矛盾であることを確かめることである。解析的に計算した力と全エネルギーの差分から数値的に計算した力は誤差範囲内で一致しなければならない。37種類の入力ファイルに対して自動的に解析的力と数値的力を計算し、比較する。

オプションをつけてopenmxを実行するだけで良い。

% ./openmx -forcetest 0

最後のフラグにより、各エネルギー成分毎にチェックできる。

flag	0	1	2	3	4	5	6	7	8
Kinetic	1	0	1	1	1	1	1	1	1
Non-local	1	0	0	1	0	0	0	0	0
Neutral atom	1	0	0	0	1	0	0	0	0
diff Hartree	1	0	0	0	0	1	0	0	0
Ex-Corr	1	0	0	0	0	0	1	0	0
E. Field	1	0	0	0	0	0	0	1	0
Hubbard U	1	0	0	0	0	0	0	0	1

37種類の入力に対する比較結果はforcetest.resultに出力される。

```
[ozaki@mx63 tmp]$ more forcetest.result

force_example/C2_GGA.dat
  flag= 0
  Numerical force= -(Utot(s+ds)-Utot(s-ds))/(2*ds)
  ds=   0.0003000000
  Forces (Hartree/Bohr) on atom 1
                              x                y                z
  Analytic force     -1.675899852770 -1.396714703544 -1.117885271947
  Numerical force    -1.675841463911 -1.396807488557 -1.117790446476
  diff               -0.000058388859  0.000092785013 -0.000094825471
```

プログラム実装の品質テスト#4

小規模、中規模、大規模計算テスタ

多岐に渡る計算機プラットフォーム上でコンパイルが正常に行われ、正常に稼働する実行ファイルが生成できているのか、確認する必要がある。小規模系(１４種類)、中規模系(16種類)、大規模(7種類)の計算を予め、開発プロットフォーム上で実行し、その結果を保存しておく。各プラットフォーム上で同様な計算を自動で実行し、参照との差異を比較する。

オプションをつけてopenmxを実行するだけで良い。

% mpirun -np 8 openmx -runtest -nt 2

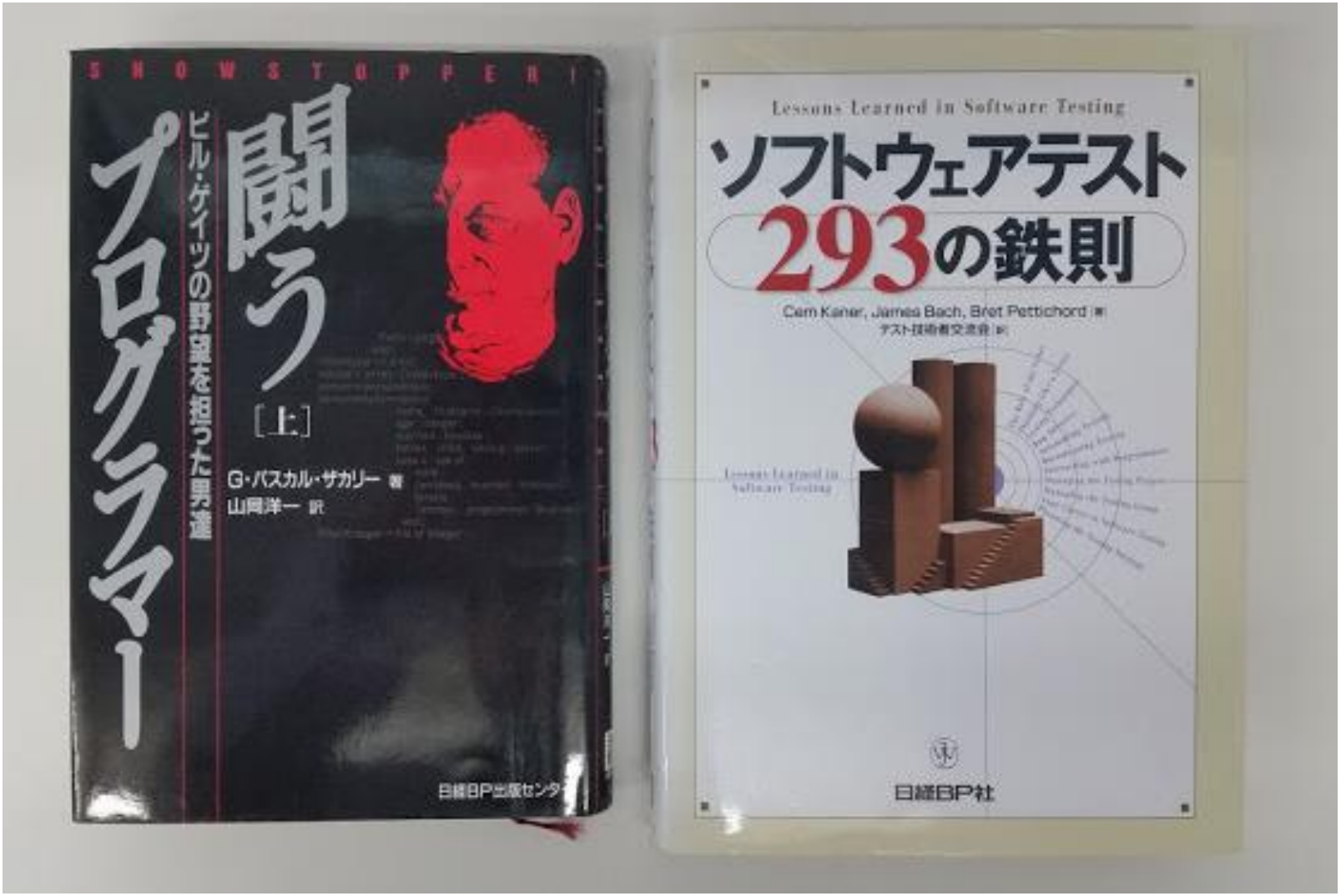
結果はruntest.resultに出力される。

```
[ozaki@mx63 work]$ more runtest.result
   1  input_example/Benzene.dat        Elapsed time(s)=    4.53   diff Utot= 0.000000000006   diff Force= 0.000000000273
   2  input_example/C60.dat            Elapsed time(s)=   14.08   diff Utot= 0.000000001575   diff Force= 0.002935619936
   3  input_example/CO.dat             Elapsed time(s)=    8.57   diff Utot= 0.000000000314   diff Force= 0.000032587000
   4  input_example/Cr2.dat            Elapsed time(s)=    7.93   diff Utot= 0.000000001907   diff Force= 0.000000000068
   5  input_example/Crys-MnO.dat       Elapsed time(s)=   16.52   diff Utot= 0.000000000006   diff Force= 0.000000000006
   6  input_example/GaAs.dat           Elapsed time(s)=   22.58   diff Utot= 0.000000000011   diff Force= 0.000000000001
   7  input_example/Glycine.dat        Elapsed time(s)=    4.85   diff Utot= 0.000000000003   diff Force= 0.000079481163
   8  input_example/Graphite4.dat      Elapsed time(s)=    4.04   diff Utot= 0.000000000000   diff Force= 0.000000000002
   9  input_example/H2O-EF.dat         Elapsed time(s)=    3.60   diff Utot= 0.000000000118   diff Force= 0.000013638495
  10  input_example/H2O.dat            Elapsed time(s)=    3.29   diff Utot= 0.000000000103   diff Force= 0.000016437469
  11  input_example/HMn.dat            Elapsed time(s)=   13.89   diff Utot= 0.000000120641   diff Force= 0.000003445768
  12  input_example/Methane.dat        Elapsed time(s)=    2.90   diff Utot= 0.000000000002   diff Force= 0.000000002786
  13  input_example/Mol_MnO.dat        Elapsed time(s)=    8.62   diff Utot= 0.000000000877   diff Force= 0.000016051151
  14  input_example/Ndia2.dat          Elapsed time(s)=    4.77   diff Utot= 0.000000000001   diff Force= 0.000000000000

Total elapsed time (s)       120.19
```

全エネルギーと力の絶対差異が出力される。また計算時間も出力されるため、
各プラットフォーム間の計算速度の比較が容易にできる。

品質テストに関して時々、参考にしている本



品質テストは兎に角、大変であることがよく分かる。F.P. Brooks, Jrの論文「銀の弾丸はない（No Silver Bullet）」が良く引き合いに出される。

ソフトウエア公開の際の心構え

- 広く一般に公開し、多数の研究者に活用されるソフトウエアの開発を目指すならば…
- 開発者の都合でなく、常にユーザーの視点に立って、開発を進める必要がある。
- 科学技術の発展のため、ひいては社会の発展のため、大きな志を持って、長期に渡る無償でのボランティアが必要である。
- ユーザーは厳しい。自分のことを振り返れば、よく分かる。私はWindows OSのユーザーであるが、文句ばかり言っている。長期に渡って批判に耐え続け、それを糧に改良をし続ける精神力が必要である。

一次情報の公開は英語で

OpenMXは、全ての一次資料を英語で公開している。

その意図は…

Web等を通してソフトウエアを一般公開した場合、ユーザーは国内に留まらない。将来の大いなる発展を考えるならば、

ソフトウエア公開に関係する情報

- マニュアル
- テクニカルノート
- フォーラム
- データベース

等は全て、英語で公開することがベターであると思われる。
日本国内の学生等のtutorial等では資料は日本語でも良いが、基本は英語で日本語資料は英語から起こすことがベター。

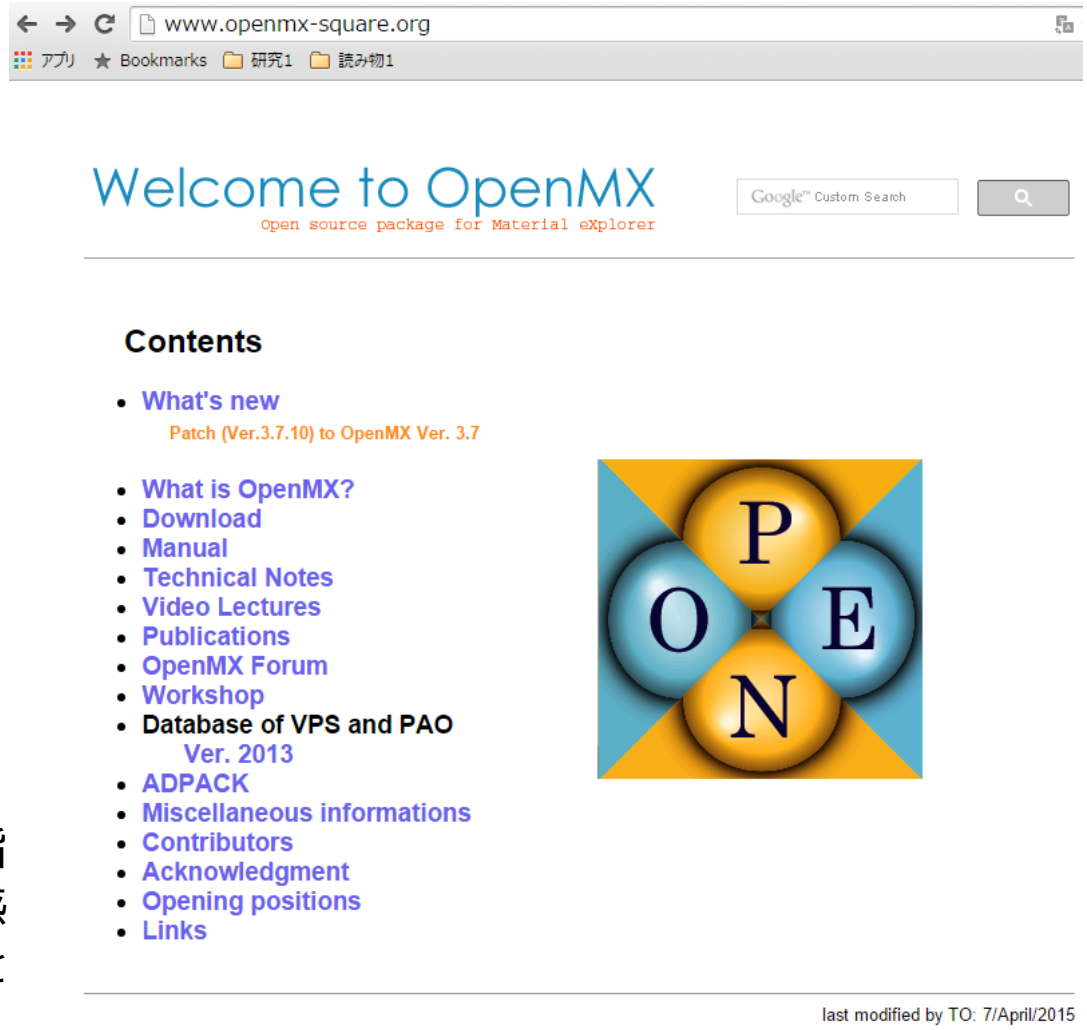
世界的に評価されたソフトウエアでないと、日本国内でシェアを取ることは出来ない。まず世界の荒波で揉まれ、その後に日本展開。その逆、「日本で揉まれ、その後に世界展開」は難しい。

公式WEBを情報共有の拠点に

計算を始めるにあたり、必要な情報を分かりやすい形で提示することが必要。

- ソフトウエア紹介
- ダウンロード
- マニュアル
- インストール情報
- テクニカルノート
- 出版論文一覧
- ユーザーフォーラム
- ワークショップ情報
- データベース
- 開発者一覧
- 謝辞　等

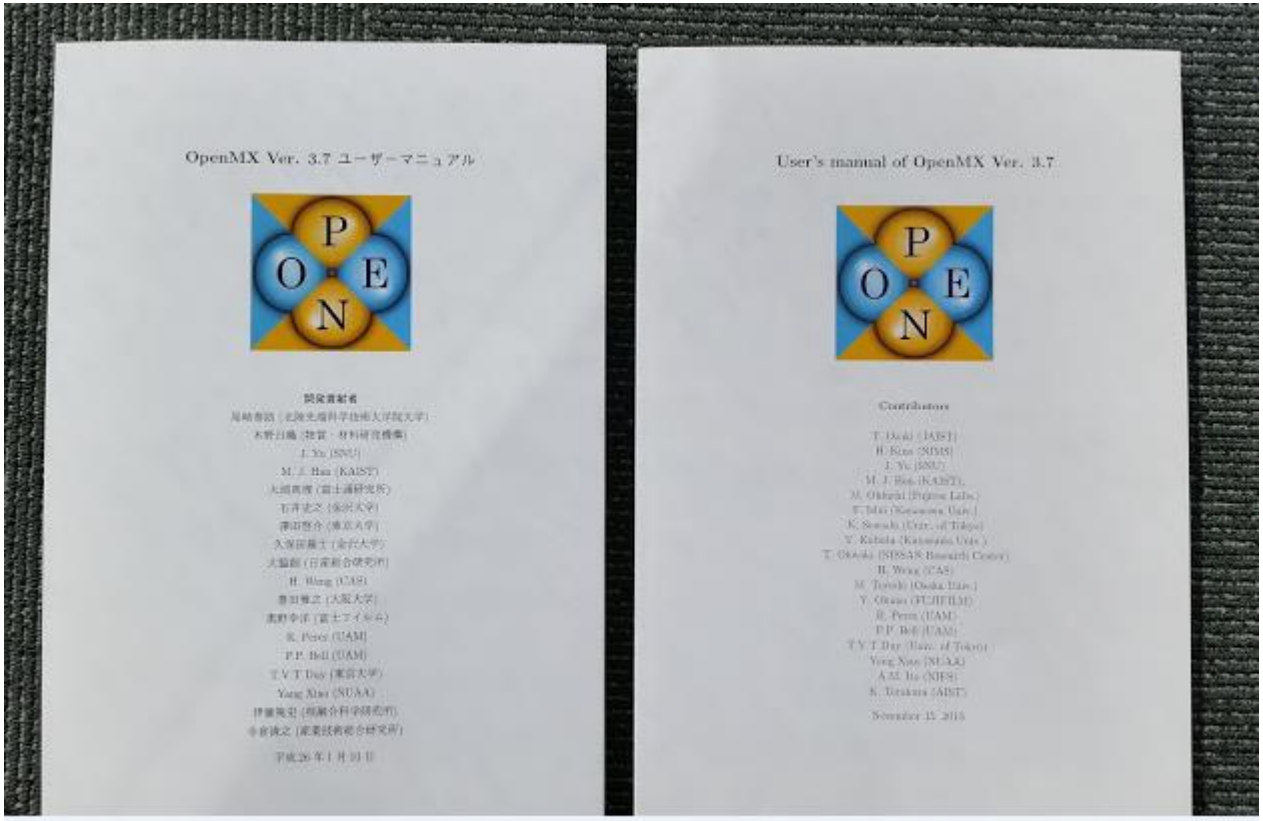
ソフトウエア開発の永続性を目指し、かつユーザーに対する安心感を獲得するため、独自ドメインを取得することが望ましい。



マニュアル作成時に留意していること

- マニュアルに記載された計算をユーザーがトレースできるようにする。
- 同じ計算を実行し、マニュアルに記載された結果と比較しながら、理解できるように配慮。
- ソフトウエア公開版にはマニュアルの計算に用いた入力ファイルも含める。
- 英語版をまず作成し、その後に日本語版を作成。

日本語版 208p　　英語版 196p



OpenMXリリースの実際

1. 情報集約

数か月前に次期リリースに関して関係各位に連絡し、寄与できる/したい機能があるか照会する。ある場合には組み込み作業を共同で実施する。

2. テスト実施

自動テスタを用いて、Intel、AMD、IBM、日立、富士通等の環境下でテストを実施し、問題がある場合には修正し、再度テストを実施する。

メモリリークテスタ、力の計算テスタ、小規模、中規模、大規模計算テスタ、構造最適化テスタ、伝導計算テスタ、ワニエ関数計算テスタ

3. マニュアル・Webの整備

新しい機能に関してマニュアルに記載。マニュアル中の計算はほぼ全てユーザーが再現できるように入力ファイルも整備。同様な内容はWeb上にも反映。

上記(2),(3)に要する時間はフルタイムで１か月程度

ユーザービリティ向上への工夫

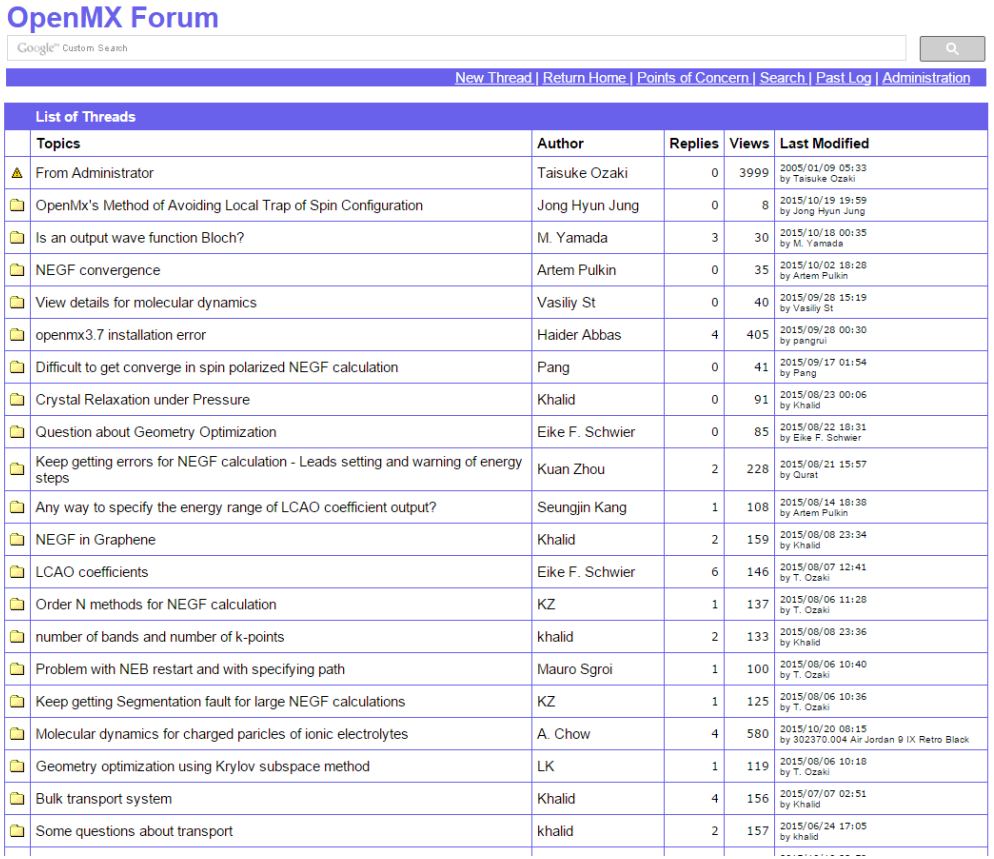
- Web上の多くの情報を蓄積。
- 75種類の元素に対して最適化基底関数と最適化擬ポテンシャルを提供
- 様々なベンチマーク計算の結果と使用した入力ファイルをweb上で公開
- Intel, AMD、富士通、日立コンパイラ、MLK, ACML等を用いた場合のコンパイル方法とトラブル対処法を公開
- 無駄な計算を避けるため、計算が異常終了した場合に対処するため、様々なリスタート機能を用意。
  - SCF計算
  - 準ニュートン法による構造最適化
  - Wannier関数の再計算
  - 透過率、電流の再計算
- on-the-flyでSCF計算のパラメーターを変更可能

ユーザーサポートの実際

- Web上でOpenMX Forumを設け、出来るだけユーザー間の情報共有を促進するように努力している。
- それでも対応が必要となる不具合に関する問い合わせは平均すると月に１、2回程度はある。
- これまでにユーザーからのリポートによって、100個レベルのバグ(軽微なもの・要望に近いものも含めて)がフィックスされている。これは公開の大きなメリットである。
- ユーザーサポートをユーザー自身が行う”スーパーユーザー”の成長を期待しているが、まだ十分とは言い難い状況。
- またユーザーサポート＆コミュニティー形成のためにこれまでに３回の国際会議を含め12回のワークショップを開催。

OpenMX Forum

- 2005年よりOpenMX Web上でOpenMX Forumを設け、不具合等の報告やユーザー間の情報共有を行っている。
- これまでに700件程度の問い合わせがあり、それへの回答等を含めて投稿数は1800件程度ある。
- ユーザー間の情報共有はかなり進んでいるが、もう少し開発者側の回答者を増やしたいと考えている。



OpenMX Forum

Google™ Custom Search

New Thread | Return Home | Points of Concern | Search | Past Log | Administration

List of Threads

	Topics	Author	Replies	Views	Last Modified
⚠	From Administrator	Taisuke Ozaki	0	3999	2005/01/09 05:33 by Taisuke Ozaki
🗀	OpenMx's Method of Avoiding Local Trap of Spin Configuration	Jong Hyun Jung	0	8	2015/10/19 19:59 by Jong Hyun Jung
🗀	Is an output wave function Bloch?	M. Yamada	3	30	2015/10/18 00:35 by M. Yamada
🗀	NEGF convergence	Artem Pulkin	0	35	2015/10/02 18:28 by Artem Pulkin
🗀	View details for molecular dynamics	Vasiliy St	0	40	2015/09/28 15:19 by Vasiliy St
🗀	openmx3.7 installation error	Haider Abbas	4	405	2015/09/28 00:30 by pangrui
🗀	Difficult to get converge in spin polarized NEGF calculation	Pang	0	41	2015/09/17 01:54 by Pang
🗀	Crystal Relaxation under Pressure	Khalid	0	91	2015/08/23 00:06 by Khalid
🗀	Question about Geometry Optimization	Eike F. Schwier	0	85	2015/08/22 18:31 by Eike F. Schwier
🗀	Keep getting errors for NEGF calculation - Leads setting and warning of energy steps	Kuan Zhou	2	228	2015/08/21 15:57 by Qurat
🗀	Any way to specify the energy range of LCAO coefficient output?	Seungjin Kang	1	108	2015/08/14 18:38 by Artem Pulkin
🗀	NEGF in Graphene	Khalid	2	159	2015/08/08 23:34 by Khalid
🗀	LCAO coefficients	Eike F. Schwier	6	146	2015/08/07 12:41 by T. Ozaki
🗀	Order N methods for NEGF calculation	KZ	1	137	2015/08/06 11:28 by T. Ozaki
🗀	number of bands and number of k-points	khalid	2	133	2015/08/08 23:36 by Khalid
🗀	Problem with NEB restart and with specifying path	Mauro Sgroi	1	100	2015/08/06 10:40 by T. Ozaki
🗀	Keep getting Segmentation fault for large NEGF calculations	KZ	1	125	2015/08/06 10:36 by T. Ozaki
🗀	Molecular dynamics for charged paricles of ionic electrolytes	A. Chow	4	580	2015/10/20 08:15 by 302370.004 Air Jordan 9 IX Retro Black
🗀	Geometry optimization using Krylov subspace method	LK	1	119	2015/08/06 10:18 by T. Ozaki
🗀	Bulk transport system	Khalid	4	156	2015/07/07 02:51 by Khalid
🗀	Some questions about transport	khalid	2	157	2015/06/24 17:05 by khalid

ユーザーサポート: Wien2kの場合

- 数年前にWien2kを購入し、メーリングリストに加入しました。
- 毎日、3－4通のメールを受信します。
- Wien2kはユーザー数も多いのでそれは驚くことではないのですが…,

- 大変に驚いたことがあります。
- 過去数年、Blaha教授がほぼ毎日、返信しているのです。
- Wien2kの開発とその普及に掛ける情熱は並大抵ではありません。

ワークショップの開催

国際ワークショップ、サマースクール、ハンズオン等、１２回のワークショップを継続的に開催。
開発者とユーザーのコミュニティ形成を期待。

2015
The 3rd OpenMX/QMAS workshop 2015 at ISSP
Hands-on workshop 2015 in ISSP

2014
Hands-on workshop 2014 in Kobe

2013
The tutorial summer school 2013 in JAIST
Seminar on OpenMX, QMAS, and TOMBO at AIST-Kansai

2012
Intensive Lectures on Advanced Computational Materials Science at Nanjing University

2011
The 1st OpenMX study meeting in JAIST
HPC nanotechnology school in Tokyo

2010
The 2nd OpenMX/QMAS workshop 2010 in Tokyo

2009
The tutorial summer school 2009 in JAIST

2008
The 1st OpenMX/QMAS workshop 2008 in JAIST

2007
Mini-school on Fast O(N) Electronic Structure Calculations: Principle and Implementations in the OpenMX package at National Taiwan Univ.

The 1st OpenMX/QMAS workshop 2008

The 2nd OpenMX/QMAS workshop 2010

The 3rd OpenMX/QMAS workshop 2015

共同開発におけるポリシー

- これまで「来るもの拒まず、去る者追わず」のポリシーで共同開発を行ってきた。
- いずれの共同開発の場合においても、共同研究者に丸投げの形は取らず、可能な限り計算手法及び実装内容を理解するように努めている。
- リリース後に該当する部分のバグ対応やメンテナンスが必要となるが、ボランティアベースで共同開発を行っている限り、リリース後の対応に対して義務を求めることは難しい。多くのケースで努力は見られるが、各研究者はそれぞれの課題を持っており、即時対応は無理な場合が多い。
- ソフトウエアの開発責任者が理解していれば、責任を持って対応にあたることができる。もし出来なければ、コードの該当機能は死んだことになる。
- 長期的な視点を持ちながら、共同開発を進めていかないと、人間関係にも遺恨を残すことになる。

共同開発の実際#1

1. 双方の同意のもとで、共同開発テーマを決定する。

2. 定式化を実施する。私が行うこともあれば、先方が実施する場合もいろいろある。

3. 2で得られた定式化と現行のプログラムのデータ構造を重ね合わせ検討し、プログラミングの大筋を共有する。

4. プログラミングの分担を決める。ケースバイケースで割り当てを決める。私が100%の場合もあれば、先方が100%の場合もある。

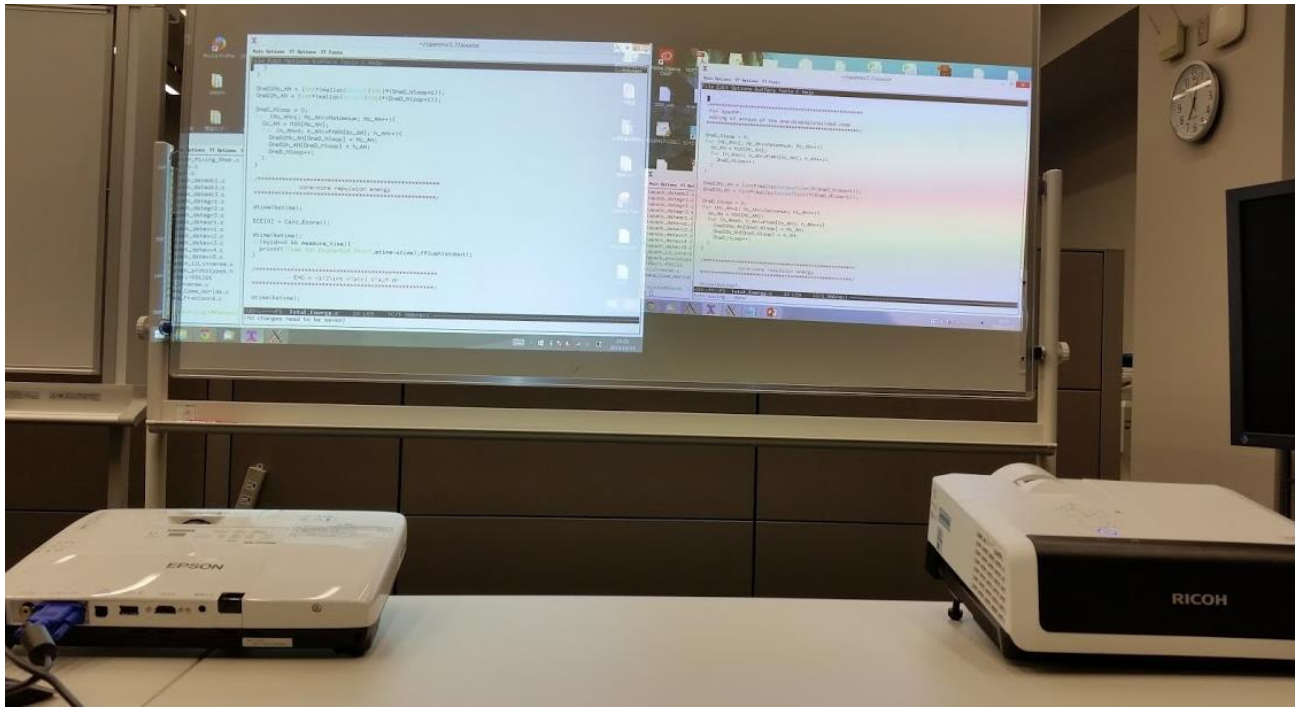
5. 実装を行い、テスト計算を実施する。先方のデバッグに私が参戦することもしばしばある。

6. 論文作成、マニュアル作成の上でリリースする。

上記の流れでノンコリニアDFT、Wannier関数、band unfolding法、NEGF、NEB、ESM、ストレスなどが共同開発された。

共同開発の実際#2

プログラムの組み込みやデバックの際には二つのプロジェクターでプログラムを並べて表示し、ペアプログラミングすると効率的である。相手の技術や使用しているツール等も分かって意外な発見がある。

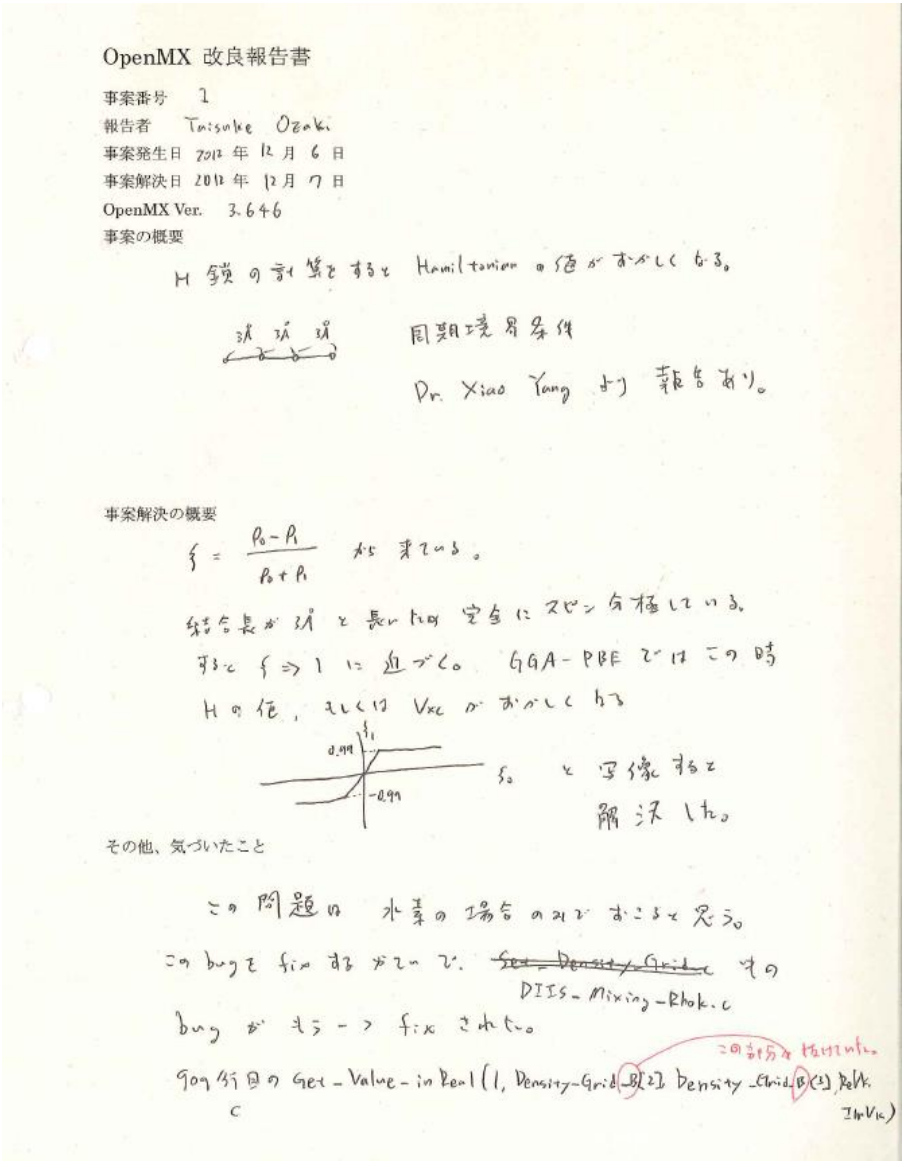


プログラムの保守管理#1

- バージョンアップは必ず特定のマシンAで行っている。
  3.7xxとしてバージョンアップする。
- 他のマシン上で見つかった問題や改良も必ずマシンA上の最新バージョンに反映させる。
- 共同研究者と最新バージョンを共有した際には、必ずマシンA上のバージョンを一つ上げる。
- 共同研究者の改良もしくは付け加えたルーチンには特定のマーカーを付けてもらい、grepで検索できるようにする。
- 共同研究者にテクニカルノート、改良ノート、参照となる入力・出力ファイルを用意してもらい、マシンA上の最新バージョンに組み込む。多くのケースでペアプログラミングを行う。

プログラムの保守管理#2

- 紙媒体で改良報告書を残すようにしている。
- 事案発生日、事案解決日、改良バージョン、事案の概要、事案解決の概要等を簡潔に記載し、ファイリングしている。
- ただし、この報告書は共同研究者間で共有していない。これに関しては今後の課題である。



OpenMX 改良報告書

事案番号 1
報告者 Tatsuke Ozaki
事案発生日 2012 年 12 月 6 日
事案解決日 2012 年 12 月 7 日
OpenMX Ver. 3.6.4.6
事案の概要

H 鎖の計算をすると Hamiltonian の値がおかしくなる。

試 試 試

周期境界条件

Dr. Xiao Yang より 報告あり。

事案解決の概要

$\zeta = \frac{\rho_0 - \rho}{\rho_0 + \rho}$ が 1 である。

結合長が 2Å と長いため 完全にスピン分極している。

すると $\zeta \Rightarrow 1$ に 近づく。 GGA-PBE では この時

H の値, もしくは Vxc が おかしくなる

と 予測すると

解決した。

その他、気づいたこと

この問題は 水素の場合のみで おこると思う。

この bug を fix するだけで、~~Set_Density_Grid~~ その

DIIS_Mixing_Rhok.c

bug が もう -> fix された。

この部分を 修正した。

709 行目の Get_Value_in_Real(1, Density_Grid_B[2] Density_Grid_B[3] RelK.

c InVk)

プログラムの保守管理#3

- バグの報告があった場合の(私が思う)ベストな対処法は可及的速やかに対応することである。
- Ver. 3.7のリリース後、すぐにユーザーから複数の不具合報告があった。バグフィックスし、速やかにパッチをリリース。これまでに10回のパッチをリリース
- 公開ソフトウエアはユーザーによって様々な計算機環境上で試験されていると言える。ユーザーからのフィードバックに対して、速やかに対応していくことで、頑健なソフトウエアとして成長していく。
- この作業を怠ると、ソフトウエアは徐々に死んでいく。

patch3.7.1: May 27, 2013
patch3.7.2: May 29, 2013
patch3.7.3: Jun. 03, 2013
patch3.7.4: Jul. 03, 2013
patch3.7.5: Aug. 29, 2013
patch3.7.6: Sep. 01, 2013
patch3.7.7: Jan. 31, 2014
patch3.7.8: Feb. 17, 2014
patch3.7.9: Feb. 20, 2015
patch3.7.10: Feb. 21, 2015

将来の発展へ向けたいくつかの展望

- 独自性の追求

  新しい計算手法開発のプラットフォームとして、今後も活用していく。単に目新しいだけでなく、シミュレーションの可能性を拡大していくような有用な手法を開発したい。

- 物質科学研究に役立つ

  新しい実験技術に基づく物質科学の研究をサポートできるシミュレーションツールとして発展させたい。スペクトル計算機能の充実が重要であろうと考えている。

- ユーザービリティの向上

  長期的な観点から物質科学のシミュレーションの発展を考えた場合、GUIの開発は必須であろう。GUIの開発はプロットフォーム依存性が大きく煩雑であり、これまで二の足を踏んでいたが、Web-based GUIに可能性を感じている。

今後の開発計画

さらなる高精度化、高速化、省メモリ化、多機能化の追求

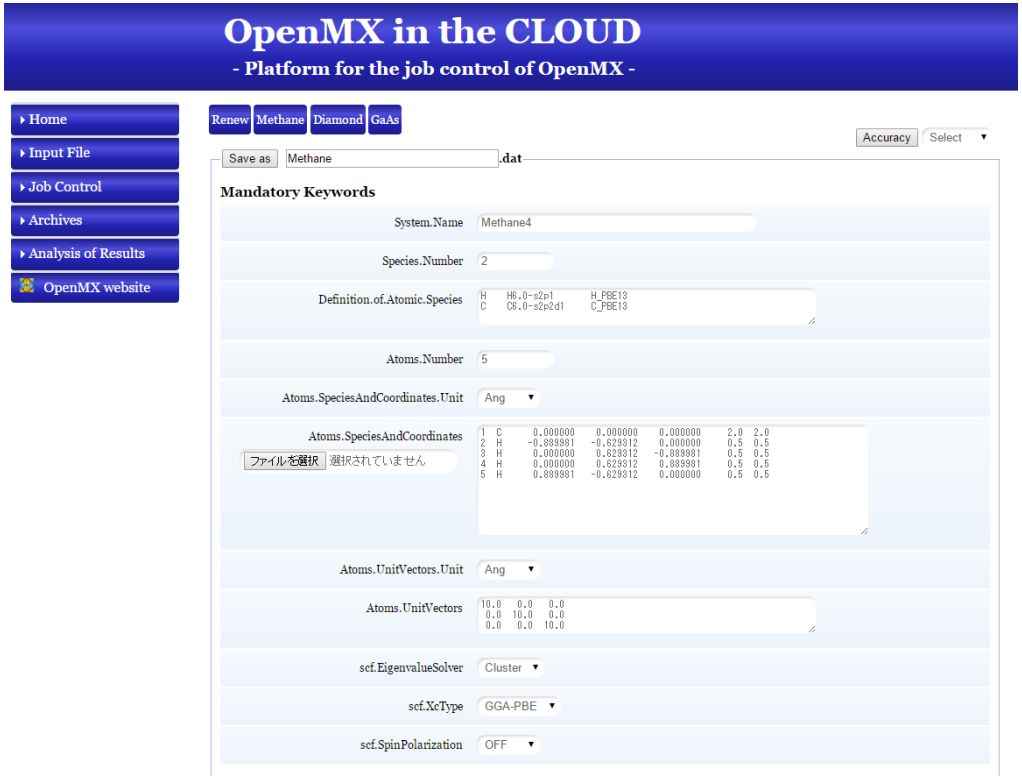
- **計算の高速化と省メモリ化**
  O(N)法、O(N^3)法、SCF計算、構造最適化
- **スペクトル計算**
  フォノン分散、unfolding法、可視紫外吸収、
  IR、Raman, XAS、NMR、ESR
- **ストレス計算**
  ストレス、局所ストレス、局所エネルギー解析
- **GGAを越える方法論開発**
  ハイブリッド汎関数、vdW-DF法、mBJ法、
  機械学習汎関数、GW法、RPA法
- **電気伝導計算の多機能化**
  多端子計算、ゲート電圧、**固有チャネル解析、電流密度解析、**
  O(N)NEGF計算、フォノン散乱、ノンコリニア、スピン軌道相互作用
- **分子動力学計算の多機能化**
  Constant μ-ESM法、 Blue moon法、ニューラルネットワーク法

赤字: 取組中
黒字: 計画中
青字: 物性研
ソフトウエア
高度化支援

Web-based GUIの開発

第一原理電子状態計算のユーザーの裾野は広がっている。実験家も「測定装置」の一つとして、積極的に活用している。将来のさらなる普及を考えるとユーザーフレンドリーなGUIの提供は必須であろう。GUIの開発はプラットフォーム依存性が大きく、多くの研究者は躊躇しているように見える。Web-based GUIはその問題を解決できる可能性がある。

HTML5, CSS3, Javascript, CGI, jQuery, WebGL等はほぼ全てのブラウズがサポートしており、ユーザーはダウンロード、インストール、プラグイン等なしで、簡単に計算を行うことができる。



ソフトウエア開発は長期戦

- “あまりやりたくないこと”でも、プロジェクト研究で2－5年は頑張れますが、20年間は普通、頑張れません。
- デファクトスタンダードソフトウエアの多くは20年以上の開発期間を要しています。
- ソフトウエア開発を”あまりやりたくないこと”から”ぜひやりたいこと”に発想を変えない限り、世界に伍していくソフトウエアを開発することは難しいのではないかと思います。

ソフトウエアを育てるために

日本発の科学技術ソフトウエアを育てるためには二つの因子が重要だと考えます。

(1) チャレンジを許す環境

若い研究者がソフトウエア開発を研究主題と出来る環境を整えることが重要。結果として、多くの試みは失敗に終わる可能性が高いが、そこから新たな芽が出てくる。その可能性の土壌を豊かにしておくことが肝要である。

(2) 芽を伸ばす環境

多くのチャレンジの中から芽が出て来て伸びる可能性のある試みを取り上げ、サポートする体制が必要である。デファクトスタンダードソフトウエアの多くは20年以上の開発期間を要している。長期的な視点からソフトウエア開発をサポートすることが肝要である。

「次世代スーパーコンピューティング・シンポジウム2007」
－ ペタスケール・システムの利用に向けて －

で述べた所感

オープンソースコード開発における所感

正の効用

知識の共有化

知識の融合化

$\frac{\text{仕事量}}{N}$　　N: 人数

負の効用

コミュニケーションの増大: αN^2

GNU-GPL的な自由スタイルを取りつつも、現実は開発責任者がコードの整合性、手法開発の妥当性等の責任を払う必要がある。現実的な解は外との連携を保った家内工業的な教育・開発体制。

OpenMXとDFT: Agenda②

- 安定性向上へのいくつかの工夫
- プログラム実装の品質テスト
- プログラム公開の際の心構え
- 公開は英語で
- マニュアル作成時に留意していること
- リリースの実際
- ユーザービリティ向上への工夫
- ユーザーサポートの実際
- ワークショップ・講習会の開催
- 共同開発におけるポリシー
- 共同開発の実際
- プログラムの保守管理
- 将来の発展へ向けたいくつの展望
- ソフトウエア開発は長期戦
- ソフトウエアを育てるために